On Constructions of Involutory MDS Matrices

Kishan Chand Gupta and Indranil Ghosh Ray

Applied Statistics Unit, Indian Statistical Institute,
203, B.T. Road, Kolkata 700108, India
{kishan,indranil_r}@isical.ac.in

Abstract. Maximum distance separable (MDS) matrices have applications not only in coding theory but also are of great importance in the design of block ciphers and hash functions. It is highly nontrivial to find MDS matrices which is involutory and efficient. In a paper in 1997, Youssef et. al. proposed an involutory MDS matrix construction using Cauchy matrix. In this paper we study properties of Cauchy matrices and propose generic constructions of low implementation cost MDS matrices based on Cauchy matrices. In a 2009 paper, Nakahara and Abrahao proposed a 16×16 involutory MDS matrix over $\mathbb{F}_{2^8}$ by using a Cauchy matrix which was used in MDS-AES design. Authors claimed that their construction by itself guarantees that the resulting matrix is MDS and involutory. But the authors didn't justify their claim. In this paper we study and prove that this proposed matrix is not an MDS matrix. Note that this matrix has been designed to be used in the block cipher MDS-AES, which may now have severe weaknesses. We provide an algorithm to construct involutory MDS matrices with low Hamming weight elements to minimize primitive operations such as exclusive-or, table look-ups and xtime operations. In a 2012 paper, Sajadieh et. al. provably constructed involutory MDS matrices which were also Hadamard in a finite field by using two Vandermonde matrices. We show that the same matrices can be constructed by using Cauchy matrices and provide a much simpler proof of their construction.

Keywords: Cauchy matrix, Diffusion, Involutory matrix, MDS matrix, MixColumn operation, Vector space, Subspace, Vandermonde matrix.

1 Introduction

Claude Shannon, in his paper "Communication Theory of Secrecy Systems" [24], defined *confusion* and *diffusion* as two properties, required in the design of block ciphers. One possibility of formalizing the notion of perfect diffusion is the concept of *multipermutation*, which was introduced in [23, 26]. Another way to define it is using MDS matrices. *Maximum Distance Separable (MDS) matrices* offer diffusion properties and is one of the vital constituents of modern age ciphers like Advanced Encryption Standard (AES) [6], Twofish [21, 22], SHARK [18], Square [5], Khazad [1], Clefia [25] and MDS-AES [10]. The stream cipher MUGI [27] uses MDS matrix in its linear transformations. MDS matrices

A. Youssef, A. Nitaj, A.E. Hassanien (Eds.): AFRICACRYPT 2013, LNCS 7918, pp. 43–60, 2013.

are also used in the design of hash functions. Hash functions like Maelstrom [7], Grϕstl [8] and PHOTON family of light weight hash functions [9] use MDS matrices as main part of their diffusion layers.

Nearly all the ciphers use predefined MDS matrices for incorporating the diffusion property. Although in some ciphers the possibility of random selection of MDS matrices with some constraints is provided [30]. In this context we would like to mention that in papers [9, 11, 14, 19, 30], different constructions of MDS matrices are provided. In [9], authors constructed lightweight MDS matrices from *companion matrices* by exhaustive search. In [11], authors constructed efficient 4×4 and 8×8 matrices to be used in block ciphers. In [14, 19], authors constructed *involutory* MDS matrices using *Vandermonde matrices*. In [30], authors constructed new involutory MDS matrices using properties of *Cauchy matrices*.

There are two very popular approaches for the design of large MDS matrices. One involves Cauchy matrices [30] and the other uses Vandermonde matrices [14, 19]. In some recent works [9, 20, 29], MDS matrices have been constructed recursively from some suitable companion matrices for lightweight applications.

In [28], authors proposed a special class of *substitution permutation networks (SPNs)* that uses same network for both the encryption and decryption operations. The idea was to use *involutory* MDS matrix for incorporating diffusion. It may be noted that for ciphers like FOX [12] and WIDEA-n [13] that follow the Lai-Massey scheme, there is no need of involutory matrices.

In this paper we revisit and systematize the MDS matrix constructions using Cauchy matrices [30] and generalize it. We also study involutory MDS matrices where the entries are preferably of low *Hamming weight*.

Lacan and Fimes [14] constructed MDS matrices from two Vandermonde matrices. Sajadieh et. al. [19] constructed MDS matrices which were also involutory. They [19] also constructed involutory *Hadamard* MDS matrices in a finite field. In this paper we propose a Cauchy based MDS matrix construction and prove that this is Hadamard in the finite field. We further provide an interesting equivalence of our Cauchy based construction and the Vandermonde based "Hadamard involutory MDS matrix" construction of [19]. By this equivalence we have a much simpler proof of generalization of Corollary 2 of [19]. We also show that our method is faster than the Hadamard involutory MDS matrix construction of [19] in terms of time complexity.

In [10], authors proposed a new diffusion layer for their AES cipher that may replace the original *ShiftRow* and *MixColumn* layers. They proposed a new 16×16 matrix $M_{16\times16}$ for designing MDS-AES block cipher, which was claimed to be involutory and MDS. But the authors did not justify their claims. In this paper we prove that their claim is not correct and the constructed $M_{16\times16}$ matrix is not an MDS matrix. Our construction (Algorithm 2) may be used to generate 16×16 involutory MDS matrices which may be used in MDS-AES block cipher.

MDS matrices of low Hamming weight are desirable for efficient implementation. In this context it may be noted that multiplication by 1, which is the unit element of $\mathbb{F}_{2^n}$, is trivial. When α is the root of the constructing polynomial of $\mathbb{F}_{2^n}$, the multiplication by α can be implemented by a shift by one bit to the left

and a conditional XOR with a constant when a carry bit is set (multiplication by α is often denoted as xtime). Multiplication by $\alpha+1$ is done by a multiplication by α and one XOR operation. Multiplication by α^2 is done by two successive multiplications by α.

The organization of the paper is as follows: In Section 2 we provide definitions and preliminaries. In Section 3, we construct MDS matrices using Cauchy matrices. In Section 4 we study Cauchy and Vandermonde constructions for FFHadamard involutory MDS matrices. In Section 5 we show that the 16×16 matrix $M_{16\times 16}$ as proposed in [10] is not MDS. We conclude the paper in Section 6.

2 Definition and Preliminaries

Let $\mathbb{F}_2 = \{0, 1\}$ be the finite field of two elements and $\mathbb{F}_{2^n}$ be the finite field of 2^n elements. Elements of $\mathbb{F}_{2^n}$ can be represented as polynomials of degree less than n over $\mathbb{F}_2$. For example, let $\beta \in \mathbb{F}_{2^n}$, then β can be represented as $\sum_{i=0}^{n-1} b_i\alpha^i$, where $b_i \in \mathbb{F}_2$ and α is the root of generating polynomial of $\mathbb{F}_{2^n}$. Another compact representation uses hexadecimal digits. Here the hexadecimal digits are used to express the coefficients of corresponding polynomial representation. For example $\alpha^7+\alpha^4+\alpha^2+1 = 1.\alpha^7+0.\alpha^6+0.\alpha^5+1.\alpha^4+0.\alpha^3+1.\alpha^2+0.\alpha+1 = (10010101)_2 = 95_x \in \mathbb{F}_{2^8}$. We will often denote a matrix by $((a_{i,j}))$, where $a_{i,j}$ is the (i, j)-th element of the matrix.

The Hamming weight of an integer i is the number of nonzero coefficients in the binary representation of i and is denoted by $\mathrm{H}(i)$. For example $\mathrm{H}(5) = 2$, $\mathrm{H}(8) = 1$.

$\mathbb{F}_{2^n}$ and $\mathbb{F}_2^n$ are isomorphic when both of them are regarded as *vector space* over $\mathbb{F}_2$. The isomorphism is given by $x = (x_1\alpha_1 + x_2\alpha_2 + \cdots + x_n\alpha_n) \mapsto (x_1, x_2 \cdots, x_n)$, where $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ is a basis of $\mathbb{F}_{2^n}$.

Let $(H, +)$ be a *group* and G is a *subgroup* of $(H, +)$ and $r \in H$. Then $r + G = \{r + g : g \in G\}$ is *left coset* of G in H and $G + r = \{g + r : g \in G\}$ is *right coset* of G in H. If the operation $+$ in H is commutative, $r + G = G + r$, i.e. left coset is same as right coset, and $r + G$ is simply called coset of G in H. It follows that any two left cosets (or right cosets) of G in H are either identical or disjoint.

Definition 1. *Let $\mathbb{F}$ be a finite field and p and q be two integers. Let $x \to M \times x$ be a mapping from $\mathbb{F}^p$ to $\mathbb{F}^q$ defined by the $q \times p$ matrix M. We say that it is an MDS matrix if the set of all pairs $(x, M \times x)$ is an MDS code, i.e. a linear code of dimension p, length $p + q$ and minimal distance $q + 1$.*

An MDS matrix provides diffusion properties that have useful applications in cryptography. The idea comes from coding theory, in particular from maximum distance separable code (MDS). In this context we state two important theorems from coding theory.

Theorem 1. *[16, page 33] If C is an $[n, k, d]$ code, then $n - k \geq d - 1$.*

Codes with $n-k=d-1$ are called maximum distance separable code, or MDS code for short.

Theorem 2. *[16, page 321] An $[n,k,d]$ code C with generator matrix $G=[I|A]$, where A is a $k\times(n-k)$ matrix, is MDS if and only if every square submatrix (formed from any i rows and any i columns, for any $i=1,2,\ldots,min\{k,n-k\}$) of A is nonsingular.*

The following fact is another way to characterize an MDS matrix.

Fact: 1 *A square matrix A is an MDS matrix if and only if every square submatrices of A are nonsingular.*

The following fact is immediate from the definition.

Fact: 2 *All square submatrices of an MDS matrix are MDS.*

One of the elementary row operations on matrices is multiplying a row of a matrix by a scalar except zero. MDS property remains invariant under such operations. So we have the following fact.

Fact: 3 *If A is an MDS matrix over $\mathbb{F}_{2^n}$, then A', obtained by multiplying a row (or column) of A by any $c\in\mathbb{F}_{2^n}^*$ is MDS.*

Fact: 4 *If A is an MDS matrix over $\mathbb{F}_{2^n}$, then $c.A$ is MDS for any $c\in\mathbb{F}_{2^n}^*$.*

Recall that many modern block ciphers use MDS matrices as a vital constituent to incorporate diffusion property. In general two different modules are needed for encryption and decryption operations. In [28], authors proposed a special class of SPNs that uses same network for both the encryption and decryption operation. The idea was to use involutory MDS matrices for incorporating diffusion.

Definition 2. *A matrix A is called involutory matrix if it satisfies the condition $A^2=I$, i.e. $A=A^{-1}$.*

Several design techniques have been used in past for constructing MDS matrices including exhaustive search for small matrices. For large MDS matrices, the designers prefer the following two methods: One method involves Cauchy matrices [30] and the other method uses Vandermonde matrices [14,19]. In this paper we study construction of involutory MDS matrices using Cauchy matrices. Before going into the construction, we discuss Cauchy matrix and its properties which are of special importance in our constructions.

Definition 3. *Given $x_0,x_1\ldots,x_{d-1}\in\mathbb{F}_{2^n}$ and $y_0,y_1\ldots,y_{d-1}\in\mathbb{F}_{2^n}$, such that $x_i+y_j\neq 0$ for all $0\leq i,j\leq d-1$, then the matrix $A=((a_{i,j}))$, $0\leq i,j\leq d-1$ where $a_{i,j}=\frac{1}{x_i+y_j}$ is called a Cauchy matrix [16,30].*

It is known that

$$det(A)=\frac{\prod_{0\leq i<j\leq d-1}(x_j-x_i)(y_j-y_i)}{\prod_{0\leq i,j\leq d-1}(x_i+y_j)}.$$

So provided x_i's are distinct and y_j's are distinct and $x_i+y_j\neq 0$ for all $0\leq i,j\leq d-1$, $det(A)\neq 0$, i.e. A is nonsingular. So we have the following result.

Fact: 5 *For distinct* $x_0, x_1 \ldots, x_{d-1} \in \mathbb{F}_{2^n}$ *and* $y_0, y_1 \ldots, y_{d-1} \in \mathbb{F}_{2^n}$*, such that* $x_i + y_j \neq 0$ *for all* $0 \leq i, j \leq d-1$*, the Cauchy matrix* $A = ((a_{i,j}))$, $0 \leq i, j \leq d-1$ *where* $a_{i,j} = \frac{1}{x_i + y_j}$*, is nonsingular.*

From the definition of a Cauchy matrix we have the following fact.

Fact: 6 *Any square submatrix of a Cauchy matrix is a Cauchy matrix.*

From Fact 5 and Fact 6; and for distinct x_i's and y_j's, such that $x_i + y_j \neq 0$, all square submatrices of a Cauchy matrix are nonsingular. This leads to an MDS matrix construction [30]. Towards this we have the following Lemma, which we call a *Cauchy construction*.

Lemma 1. *For distinct* $x_0, x_1 \ldots, x_{d-1}$ *and* $y_0, y_1 \ldots, y_{d-1}$*, such that* $x_i + y_j \neq 0$ *for all* $0 \leq i, j \leq d-1$*, the matrix* $A = ((a_{i,j}))$*, where* $a_{i,j} = \frac{1}{x_i + y_j}$ *is an MDS matrix.*

Proof. It is to be noted that the matrix A is a Cauchy matrix. Also from Fact 6, all of its submatrices are Cauchy matrices. Since $x_0, x_1 \ldots, x_{d-1}$ and $y_0, y_1 \ldots, y_{d-1}$ are distinct and $x_i + y_j \neq 0$ for all $0 \leq i, j \leq d-1$, so from Fact 5, all square submatrices of A are nonsingular. So A is an MDS matrix. □

Lemma 2. *Each row(or each column) of the* $d \times d$ *MDS matrix A, formed using construction of Lemma 1 has d distinct elements.*

Proof. The elements of i'th row of A are $\frac{1}{x_i + y_j}$ for $j = 0, \ldots, d-1$. Now $\frac{1}{x_i + y_{j_1}} = \frac{1}{x_i + y_{j_2}}$ for any two $j_1, j_2 \in \{0, \ldots, d-1\}$ such that $j_1 \neq j_2$ implies $y_{j_1} = y_{j_2}$, which is a contradiction to the fact that y_j's are distinct. Since i is arbitrary, the result holds for all rows of A. The proof for columns are similar. □

Corollary 1. *The* $d \times d$ *MDS matrix A, formed using construction of Lemma 1 has at least d distinct elements.*

Definition 4. *[16, 19] The matrix*

$$V = van(v_0, \ldots, v_{d-1}) = \begin{pmatrix} 1 & v_0 & v_0^2 & v_0^3 & \ldots & v_0^{d-1} \\ 1 & v_1 & v_1^2 & v_1^3 & \ldots & v_1^{d-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & v_j & v_j^2 & v_j^3 & \ldots & v_j^{d-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & v_{d-1} & v_{d-1}^2 & v_{d-1}^3 & \ldots & v_{d-1}^{d-1} \end{pmatrix}$$

is called a Vandermonde matrix, where v_i*'s are from any finite or infinite field.*

Fact: 7 $det(V) = \prod_{i<j}(v_i - v_j)$*, which is non zero if and only if the* v_i*'s are distinct.*

In [14], authors proposed MDS matrix construction from Vandermonde matrices, which we call a *Vandermonde construction*. We record this important result in the following lemma.

Lemma 3. *[14, 19] For distinct $x_0, x_1, \ldots, x_{d-1}$ and $y_0, y_1, \ldots, y_{d-1}$, such that $x_i + y_j \neq 0$, the matrix AB^{-1} is an MDS matrix, where $A = van(x_0, \ldots, x_{d-1})$ and $B = van(y_0, \ldots, y_{d-1})$.*

Authors of [19] proposed techniques to produce involutory MDS matrices. We record this in the following lemma with slightly different notations.

Lemma 4. *[19] Let $A = van(x_0, \ldots, x_{d-1})$ and $B = van(y_0, \ldots, y_{d-1})$ are $d \times d$ invertible Vandermonde matrices in $\mathbb{F}_{2^n}$ satisfying $x_i = y_i + r$ and $x_i \neq y_j, i, j \in \{0, \ldots, d-1\}$, $r \in \mathbb{F}_{2^n}^*$, then AB^{-1} is an involutory MDS matrix.*

In [19], authors constructed a special form of MDS matrices called *Finite Field Hadamard* matrices, which is defined as follows:

Definition 5. *[2, 19] A $2^m \times 2^m$ matrix H is Finite Field Hadamard matrix (FFHadamard) in $\mathbb{F}_{2^n}$ if it can be represented as follows:*

$$H = \begin{pmatrix} U & V \\ V & U \end{pmatrix},$$

where the two submatrices U and V are also FFHadamard.

Fact: 8 *[19] Let $H = ((h_{i,j}))$ be a $2^m \times 2^m$ matrix whose first row is $(x_0\ x_1 \ldots x_{2^m-1})$ and $h_{i,j} = x_{i \oplus j}$, then H is FFHadamard and is denoted by $H = had(x_0, \ldots, x_{2^m-1})$.*

Let $H = ((h_{i,j})) = had(x_0, \ldots, x_{2^m-1})$, where $x_i \in \mathbb{F}_{2^n}$ for $i \in \{0, \ldots, 2^m - 1\}$. Then clearly $H' = ((h'_{i,j}))$ is FFHadamard, where $h'_{i,j} = r + h_{i,j}$, $r \in \mathbb{F}_{2^n}$. Also if $r + x_i \neq 0$ for $i \in \{0, \ldots, 2^m - 1\}$, then it is easy to check that the matrix $H'' = ((h''_{i,j}))$, where $h''_{i,j} = \frac{1}{h'_{i,j}}$ is also FFHadamard. We now provide Fact 9 which will be used in Theorem 4.

Fact: 9 *[19] Let $G = \{x_0, \ldots, x_{2^m-1}\}$ be an additive subgroup of $\mathbb{F}_{2^n}$, where $x_0 = 0$ and $x_i + x_j = x_{i \oplus j}$. Let $H = ((h_{i,j}))$ be a $2^m \times 2^m$ matrix over $\mathbb{F}_{2^n}$, where $h_{i,j} = \frac{1}{r + x_{i \oplus j}}$, $r \in \mathbb{F}_{2^n} \setminus G$, then H is FFHadamard.*

In [19], authors defined *Special Vandermonde matrix (SV matrix)*, which we restate differently and equivalently.

Definition 6. *Let G be an additive subgroup of $\mathbb{F}_{2^n}$ of order 2^m, which is a linear span of m linearly independent elements $\{x_1, x_2, x_{2^2}, \ldots, x_{2^{m-1}}\}$ such that $x_i = \sum_{k=0}^{m-1} b_k x_{2^k}$, where $(b_0, b_1, \ldots, b_{m-1})$ is the binary representation of i. A Vandermonde matrix $van(y_0, \ldots, y_{2^m-1})$ is called a Special Vandermonde matrix (SV matrix) if $y_i = r + x_i$, where $r \in \mathbb{F}_{2^n}$.*

We restate the generalization of Corollary 2 of [19] in the following lemma.

Lemma 5. *[19] Let $A = van(x_0, \ldots, x_{2^m-1})$ and $B = van(y_0, \ldots, y_{2^m-1})$ are Special Vandermonde matrices in $\mathbb{F}_{2^n}$, where $y_i = x_0 + y_0 + x_i$ and $y_0 \notin \{x_0, \ldots, x_{2^m-1}\}$, then AB^{-1} is an FFHadamard involutory MDS matrix.*

The proof of Corollary 2 of [19] is several pages long. In Section 4 Theorem 5, we propose an alternative and a much simpler proof.

3 Construction of MDS and Involutory MDS Matrices

From Corollary 1, a $d \times d$ matrix constructed using Lemma 1 has at least d distinct elements. In this paper we construct $d \times d$ MDS matrices with exactly d distinct elements. It has two-fold advantage. Firstly, we have to find only d elements of our liking (say of low implementation cost) to form the MDS matrix using Cauchy construction. Secondly, for construction of efficient MDS matrices, it may be desirable to have minimum number of distinct entries to minimize the implementation overheads(See [11]).

Lemma 6. *Let $G = (x_0, x_1, \ldots, x_{d-1})$ be an additive subgroup of $\mathbb{F}_{2^n}$. Let us consider the coset $r+G$, $r \notin G$ of G having elements $y_j = r+x_j$, $j = 0, \ldots, d-1$. Then the $d \times d$ matrix $A = ((a_{i,j}))$, where $a_{i,j} = \frac{1}{x_i+y_j}$, for all $0 \leq i, j \leq d-1$ is an MDS matrix.*

Proof. We first prove that $x_i + y_j \neq 0$ for all $0 \leq i, j \leq d-1$. Now, $x_i + y_j = x_i + r + x_j = r + x_i + x_j \in r + G$. But $0 \notin r + G$ (as $r \notin G$ and $0 \in G$). So $x_i + y_j \neq 0$ for all $0 \leq i, j \leq d-1$. Also all x_i's are distinct elements of the group G and y_j's are distinct elements of the coset $r + G$. Thus from Lemma 1, A is an MDS matrix. □

Remark 1. Lemma 6 gives MDS matrix of order d, where d is a power of 2. When d is not a power of 2, the construction of $d \times d$ MDS matrices over $\mathbb{F}_{2^n}$ ($d < 2^{n-1}$) is done in two steps. Firstly we construct $2^m \times 2^m$ MDS matrix A' over $\mathbb{F}_{2^n}$, where $2^{m-1} < d < 2^m$, using Lemma 6. In the next step, we select $d \times d$ submatrix A of A' of our liking (select d rows and d columns).

Fact: 10 *Lemma 3 of [30] is a particular case of Lemma 6 of this paper.*

Corollary 2. *The matrix A of Lemma 6 is symmetric.*

Proof. From definition, $a_{i,j} = a_{j,i} = \frac{1}{r+x_i+x_j}$ for all $0 \leq i, j \leq d-1$. Thus A is symmetric matrix. □

Lemma 7. *The $d \times d$ matrix A of Lemma 6 has exactly d distinct entries.*

Proof. In the ith row the elements are $a_{i,j} = \frac{1}{r+x_i+x_j}$ for $j = 0, 1, \ldots, d-1$. Since x_j's form the additive group G, $x_i + x_j$ for $j = 0, 1, \ldots, d-1$ gives all d distinct elements of G for a fixed i. Thus $r + x_i + x_j$ for $j = 0, 1, \ldots, d-1$ gives all d distinct elements of $r + G$. Since i is arbitrary, therefore in each row of A, there are d distinct elements. Since these elements are nothing but the multiplicative inverse of elements of $r + G$ in $\mathbb{F}_{2^n}$, the matrix A has exactly d different elements. □

Corollary 3. *By Lemma 2 and Lemma 7, it is evident that all rows of matrix A constructed by Lemma 6 are the permutations of the first row of A.*

Lemma 6 provides construction of MDS matrices. These matrices may not be involutory. In general, in substitution permutation networks (SPN) decryption needs inverse of A. If A is a low implementation-cost MDS matrix, then it is desirable that $A = A^{-1}$, otherwise implementation of A^{-1} may not be efficient. So we may like to make our MDS matrix to be involutory. Towards this we study the following Lemma which is also given in [30], but in a slightly different setting.

Lemma 8. *Let $A = ((a_{i,j}))$ be the $d \times d$ matrix formed by Lemma 6. Then $A^2 = c^2 I$, where $c = \sum_{k=0}^{d-1} \frac{1}{r+x_k}$.*

Proof. Let $A^2 = H = ((h_{i,j}))$. From Corollary 2, A is symmetric matrix. Therefore $h_{i,j}$ is the inner product of i'th row and j'th row of A. Therefore $h_{i,i} = \sum_{l=0}^{d-1} \frac{1}{(r+x_i+x_l)^2} = \sum_{k=0}^{d-1} \frac{1}{(r+x_k)^2} = c^2$ as x_i's and x_l's are elements of a group which is a subgroup of $\mathbb{F}_{2^n}$ of characteristic 2. Similarly for $i \neq j$, $h_{i,j} = \sum_{k=0}^{d-1} \frac{1}{(r+x_i+x_k)(r+x_j+x_k)} = \frac{1}{x_i+x_j} \sum_{k=0}^{d-1} \frac{1}{(r+x_i+x_k)} + \frac{1}{(r+x_j+x_k)} =$
$\frac{1}{x_i+x_j} \left(\sum_{k=0}^{d-1} \frac{1}{(r+x_i+x_k)} + \sum_{k=0}^{d-1} \frac{1}{(r+x_j+x_k)} \right) =$
$\frac{1}{x_i+x_j} \left(\sum_{l=0}^{d-1} \frac{1}{r+x_l} + \sum_{l'=0}^{d-1} \frac{1}{r+x'_l} \right)$. Since $\{r + x_l : l = 0, \ldots, d-1\} = r + G$ and we are working on a field $\mathbb{F}_{2^n}$ of characteristic 2, therefore $\left(\sum_{l=0}^{d-1} \frac{1}{r+x_l} + \sum_{l'=0}^{d-1} \frac{1}{r+x'_l} \right) = 0$. So $h_{i,j} = 0$. Thus $A^2 = c^2 I$. □

Corollary 4. *The matrix A of Lemma 6 is involutory if the sum of the elements of any row is 1.*

Proof. The sum of elements of any row of A is equal to $\sum_{i=0}^{d-1} \frac{1}{r+x_i} = c$, where c is as defined in Lemma 8. So if $c = 1$, $c^2 = 1$ and hence $A^2 = I$ (See Lemma 8). □

Corollary 5. *If $d \times d$ MDS matrix A is constructed using Lemma 6, then $\frac{1}{c}A$ is an involutory MDS matrix, where $c = \sum_{k=0}^{d-1} \frac{1}{r+x_k}$.*

Proof. From Lemma 8, $(\frac{1}{c}A)^2 = I$ and from Fact 4, $\frac{1}{c}A$ is MDS. □

Remark 2. Multiplication in $\mathbb{F}_{2^n}$ by 1 is trivial. So for implementation friendly design, it is desirable to have maximum number of 1's in MDS matrices to be used in block ciphers and hash functions. We know that each element in a $d \times d$ matrix A constructed by Lemma 6, occurs exactly d times (See Lemma 7). So in the construction of $d \times d$ matrix A by Lemma 6, maximum d number of 1's can occur in A. It is to be noted that A can be converted to have maximum number of 1's (i.e. d number of 1's) without distrurbing the MDS property just by multiplying A by inverse of one of its entries (See Fact 4). Although this will guarantee occurrence of 1's in every row, but with this technique we may not control Hamming weights of other $d-1$ elements. Also if A is an involutory MDS matrix, such conversion will disturbe the involutory property.

Remark 3. In [11], authors introduced the idea of efficient MDS matrices by maximizing the number of 1's and minimizing the number of occurrences of

other distinct elements from $\mathbb{F}_{2^n}^*$. It is to be noted that multiplication of each row of $d \times d$ MDS matrix A by inverse of the first elements of the respective rows will lead to an MDS matrix A' having all 1's in first column (See Fact 3). Again by multiplying each columns of $d \times d$ MDS matrix A' (starting from the second column) by inverse of the first elements of the respective columns will lead to an MDS matrix A'' having all 1's in first row and first column. Thus the number of 1's in this matrix is $2d - 1$. Although A'' contains maximum number of 1's that can be achieved starting from the MDS matrix A, but the number of other distinct terms in this case may be greater than $d - 1$. Also A'' will never be involutory.

3.1 Construction of Some Additive Subgroup G of $\mathbb{F}_{2^n}$

Recall that $\mathbb{F}_{2^n}$ and $\mathbb{F}_2^n$ are isomorphic when both of them are regarded as n dimentional vector space over $\mathbb{F}_2$. Any *subspace* of $\mathbb{F}_2^n$ is by definition an additive subgroup of $\mathbb{F}_{2^n}$. Let $B = \{x_0, \ldots, x_{m-1}\}$ be m linearly independent elements of $\mathbb{F}_{2^n}$. Then the linear span of B, denoted by G, is a subspace of $\mathbb{F}_2^n$ of dimension m and is an additive subgroup of $\mathbb{F}_{2^n}$. So G can be used to construct MDS matrix using Lemma 6. Also note that r in Lemma 6 can be any element of $\mathbb{F}_{2^n} \setminus G$. Our aim is to construct efficient MDS matrices. Hamming weights of the elements in the MDS matrix may decide the number of table lookups, xor and xtime operations. The higher order bits of each entries in the matrix affects the number of calls to xtime. In the construction of MDS matrices by Lemma 6, the elements of the matrices are inverses of the elements of $r+G$ (See Lemma 6). So it is desirable that multiplicative inverses of elements of $r+G$ in $\mathbb{F}_{2^n}$ must be of low Hamming weights and also all the 1's should be towards the lower order bits.

3.2 An Algorithm to Construct MDS Matrix

Based on Lemma 6, we now provide Algorithm 1 to construct $2^m \times 2^m$ MDS matrix over $\mathbb{F}_{2^n}$, where $m < n$. Algorithm 1 gives MDS matrix and when the input parameter $b_{Involutory}$ is set *true*, the Algorithm 1 gives involutory MDS matrix of order $d \times d$, where d is power of 2. When d is not a power of 2, the construction of $d \times d$ MDS matrices over $\mathbb{F}_{2^n}$ $(d < 2^{n-1})$ is done in two steps (see Remark 1). Firstly we construct $2^m \times 2^m$ MDS matrix A over $\mathbb{F}_{2^n}$ using Algorithm 1 and keeping input parameter $b_{Involutory} = false$, where $2^{m-1} < d < 2^m$. In the next step, we just select some suitable $d \times d$ submatrix A' of A of our liking (select d rows and d columns of our liking). Note that A'^2 may not be equal to $c^2 I$, where $c \in \mathbb{F}_{2^n}^*$. Although the matrix A' is MDS, it is not involutory (See Example 1).

Remark 4. The additive subgroup $G = \{x_0, \ldots, x_{2^m-1}\}$ in Algorithm 1 is constructed by the linear combination of m linearly independent elements labeled $x_1, x_2, x_{2^2} \ldots, x_{2^{m-1}}$ in Step 1. Note that for such group G, $x_i + x_j = x_{i \oplus j}$, $x_i, x_j \in G$. For such G, the constructed matrix A in Algorithm 1 is FFHadamard

Algorithm 1. Construction of $2^m \times 2^m$ MDS matrix or Involutory MDS matrix over $\mathbb{F}_{2^n}$

Input $n > 1$, the generating polynomial $\pi(x)$ of $\mathbb{F}_{2^n}$, $m < n$ and $b_{Involutory}$.
Output Outputs a $2^m \times 2^m$ MDS matrix A.

1: Select m linearly independent elements, labeled $x_1, x_2, x_{2^2} \ldots, x_{2^{m-1}}$ from $\mathbb{F}_{2^n}$;
2: Construct G, the set of 2^m elements $x_0, x_1, x_2, x_3, \ldots, x_{2^m-1}$, where $x_i = \sum_{k=0}^{m-1} b_k x_{2^k}$, for all $0 \leq i \leq 2^m - 1$, $(b_{m-1}, b_{m-1}, \ldots, b_1, b_0)$ being the binary representation if i;
3: Select some $r \in \mathbb{F}_{2^n} \setminus G$;
4: Construct $r + G$, the set of 2^m elements $y_0, y_1, y_2, y_3, \ldots, y_{2^m-1}$, where $y_i = r + x_i$ for all $0 \leq i \leq 2^m - 1$;
5: **if** $(b_{Involutory} == false)$: Construct $\frac{1}{y_i}$; **else** construct $\frac{1}{cy_i}$ for $i = 0, \ldots, d-1$ in the array ary_s, where $c = \sum_{k=0}^{d-1} \frac{1}{r+x_k}$.
6: Construct the $2^m \times 2^m$ matrix $A = ((a_{i,j}))$, where $a_{i,j} = ary\_s[k]$, where $i \oplus j = k$;
7: Set A as output;

(see Theorem 4). If the ordering is disturbed in Step 1 by labeling the elements differently, so that $x_i + x_j \neq x_{i \oplus j}$, the matrix A may not be FFHadamard, although it will be MDS. We maintain the same ordering while constructing additive subgroup in Algorithm 2. So Algorithm 2 also produces FFHadamard matrices.

Theorem 3. *Algorithm 1 generates $d \times d$ MDS or Involutory MDS matrices over $\mathbb{F}_{2^n}$ where $d = 2^m$, and the complexity is $O(d^2)$ operations in $\mathbb{F}_{2^n}$.*

Proof. The correctness of Algorithm 1 is immediate from Lemma 6, Lemma 8 and Corollary 4. In Algorithm 1, Step 1-Step 5 takes $O(d)$ operations. Step 6 takes $O(d^2)$ operations. Thus the time complexity of Algorithm 1 is $O(d^2)$. □

Example 1: Let $n = 8, d = 4, \pi(x) = x^8+x^4+x^3+x+1$ and $b_{Involutory} = false$. Set $r = 1$. Select $x_1 = \alpha^7+\alpha^3+\alpha^2$ and $x_2 = \alpha^7+\alpha^6+\alpha^5+\alpha^4+\alpha^2+\alpha+1$. Thus construct $x_0 = 0.x_1 + 0.x_2 = 0$ and $x_3 = 1.x_1 + 1.x_2 = \alpha^6+\alpha^5+\alpha^4+\alpha^3+\alpha+1$. So $y_0 = 1$, $y_1 = \alpha^7 + \alpha^3 + \alpha^2 + 1$, $y_2 = \alpha^7 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^2 + \alpha$ and $y_3 = \alpha^6 + \alpha^5 + \alpha^4 + \alpha^3 + \alpha$. So we have from Lemma 6 (as implemented in Algorithm 1)

$$A = \begin{pmatrix} 01_x & 02_x & 03_x & d0_x \\ 02_x & 01_x & d0_x & 03_x \\ 03_x & d0_x & 01_x & 02_x \\ d0_x & 03_x & 02_x & 01_x \end{pmatrix}, \frac{1}{c}A = \begin{pmatrix} 7a_x & f4_x & 8e_x & 01_x \\ f4_x & 7a_x & 01_x & 8e_x \\ 8e_x & 01_x & 7a_x & f4_x \\ 01_x & 8e_x & f4_x & 7a_x \end{pmatrix},$$

where $01_x = 1$, $02_x = \alpha$ $03_x = \alpha + 1$, $d0_x = \alpha^7 + \alpha^6 + \alpha^4$, $7a_x = \alpha^6 + \alpha^5 + \alpha^4 + \alpha^3 + \alpha$, $f4_x = \alpha^7 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^2$, $8e_x = \alpha^7 + \alpha^3 + \alpha^2 + \alpha$. Here $c = d0_x$. Note that the matrix A is MDS but not involutory and the matrix $\frac{1}{c}A$ is involutory MDS. To form a 3×3 MDS matrix, we may take a submatrix A' from A or $\frac{1}{c}A$. Let us consider the 3×3 submatrix A' of the involutory MDS matrix $\frac{1}{c}A$ of order 3. Here we take first three rows and columns of $\frac{1}{c}A$ for constructing A'. Thus we have

$$A' = \begin{pmatrix} 7a_x & f4_x & 8e_x \\ f4_x & 7a_x & 01_x \\ 8e_x & 01_x & 7a_x \end{pmatrix}.$$

Note that 2nd and 3rd row of A' are not permutations of its first row.

Remark 5. For an illustration purpose, we count the number of xtime and xor operations for the matrix A and A' of Example 1 without considering any optimization technique. The matrix A requires 9 xtimes and 3 xors each row, i.e. 36 xtimes and 12 xors for one matrix computation. Similarly the matrix A' requires 20 xtimes and 11 xors each row, i.e. 80 xtimes and 44 xors for one matrix computation.

3.3 An Algorithm to Construct Low Hamming Weight Involutory MDS Matrix

Here we present an algorithm (Algorithm 2) to construct efficient $d \times d$ involutory MDS matrices, where d is power of 2. By efficient matrix, we mean a matrix having maximum number of 1's and minimum number of other distinct elements of low Hamming weight (see Remark 3). In the construction using Lemma 6, a $d \times d$ MDS matrix A can have maximum d number of 1's and $d-1$ other distinct elements (See Lemma 7). In the iteration of Algorithm 2, we fix $r = 1$, which ensures that all diagonal elements are 1. Thus we have d number of 1's. For $d = 2^m$, we initially select m distinct elements of first row $a_{0,1}, a_{0,2}, a_{0,2^2} \ldots, a_{0,2^{m-1}}$ which are of low Hamming weight and compute $x_1, x_2, x_{2^2}, \ldots, x_{2^{m-1}}$, where $x_{2^i} = \frac{1}{a_{0,2^i}} + r$, $i = 0, \ldots, m-1$. We repeat this process by selecting different elements of next lowest possible Hamming weights unless we get m linearly independent elements $x_0, x_1, x_2, x_3, \ldots, x_{2^m-1}$. We next form G and $r + G$ and finally the matrix A using Lemma 6. If the matrix is not involutory, we repeat the process unless we get an involutory MDS matrix A.

Remark 6. Note that we can choose $m + 1$ out of 2^m elements of our liking to have low Hamming weight while constructing involutory MDS matrix using Algorithm 2. But we have no control upon the other $2^m - (m + 1)$ elements of the matrix.

Remark 7. Note that Algorithm 2 is similar to Algorithm 1 and is based on Lemma 6. The Algorithm 2 may not terminate for some conditions in Step 2. If we relax the conditions of low Hamming weight in Step 2, Algorithm 2 will eventually terminate but the time complexity is not clear and may depend upon many conditions.

Remark 8. Algorithm 2 generates $d \times d$ involutory MDS matrix over $\mathbb{F}_{2^n}$ where d is power of 2. The correctness of Algorithm 2 follows from Lemma 6, Lemma 8 and Corollary 4.

Example 2: Let $n = 8$, $d = 4$, $\pi(x) = x^8 + x^4 + x^3 + x + 1$. Set $r = 1$. Also let α be the root of $\pi(x)$. We will select $a_{0,1} = 02_x = \alpha$ and search for the element with next lowest possible Hamming weight for $a_{0,2}$ so that the corresponding values $\frac{1}{a_{0,1}} + 1 = x_1$ and $\frac{1}{a_{0,2}} + 1 = x_2$ are linearly independent

Algorithm 2. Construction of $2^m \times 2^m$ Involutory MDS matrix $((a_{i,j}))$over $\mathbb{F}_{2^n}$

Input $n > 1$, the generating polynomial $\pi(x)$ of $\mathbb{F}_{2^n}$ and $m < n$.
Output Outputs a $2^m \times 2^m$ involutory MDS matrix A.

1: Set $r = 1$;
2: Select m elements labeled $a_{0,1}, a_{0,2}, a_{0,2^2} \ldots, a_{0,2^{m-1}}$ from $\mathbb{F}^*_{2^n}$ of low Hamming weight;
3: Compute m elements labeled $x_1, x_2, x_{2^2}, \ldots, x_{2^{m-1}}$, where $x_{2^i} = \frac{1}{a_{0,2^i}} + r$, $i = 0, \ldots, m-1$;
4: Check if $x_1, x_2, x_{2^2} \ldots, x_{2^{m-1}}$ are linearly independent. If not, go to Step 2;
5: Construct G, the set of 2^m elements $x_0, x_1, x_2, x_3, \ldots, x_{2^m-1}$, where $x_i = \sum_{k=0}^{m-1} b_k x_{2^k}$, for all $0 \leq i \leq 2^m - 1$, $(b_{m-1}, b_{m-2}, \ldots, b_1, b_0)$ being the binary representation of i;
6: if $(r \in G)$ then go to Step 2;
7: Construct $r + G$, the set of 2^m elements $y_0, y_1, y_2, y_3, \ldots, y_{2^m-1}$, where $y_i = r + x_i$ for all $0 \leq i \leq 2^m - 1$;
8: Compute $c = \sum_{k=0}^{d-1} \frac{1}{y_k}$. if$(c \neq 1)$: go to step 2;
9: Construct the $2^m \times 2^m$ matrix $A = ((a_{i,j}))$, where $a_{i,j} = \frac{1}{x_i + y_j}$;
10: Set A as output;

and finally the resulting matrix is involutory MDS. If not involutory, we go for next element of higher Hamming weight for $a_{0,2}$. If no suitable candidate for $a_{0,2}$ is available, we set $a_{0,1} = 03_x = \alpha + 1$, and repeat the search of suitable candidate for $a_{0,2}$. We iterate and find the first suitable combination as $a_{0,1} = \alpha$ and $a_{0,2} = fc_x = \alpha^7 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^3 + \alpha^2$ which leads to an involutory MDS matrix. For such $a_{0,1}$, and $a_{0,2}$, we get $x_1 = \frac{1}{a_{0,1}} + 1 = \alpha^7 + \alpha^3 + \alpha^2$ and $x_2 = \frac{1}{a_{0,2}} + 1 = \alpha^7 + \alpha^6 + \alpha^3 + \alpha^2$. So we have $x_0 = 0.x_1 + 0.x_2 = 0$ and $x_3 = 1.x_1 + 1.x_2 = \alpha^6$. Thus $y_0 = 1$, $y_1 = \alpha^7 + \alpha^3 + \alpha^2 + 1$, $y_2 = \alpha^7 + \alpha^6 + \alpha^3 + \alpha^2 + 1$ and $y_3 = \alpha^6 + 1$. Finally, we get

$$A = \begin{pmatrix} 01_x & 02_x & fc_x & fe_x \\ 02_x & 01_x & fe_x & fc_x \\ fc_x & fe_x & 01_x & 02_x \\ fe_x & fc_x & 02_x & 01_x \end{pmatrix}.$$

Note that this matrix is involutory MDS. The MDS matrix A of Example 1 is more implementation friendly, but it is not involutory. Note that the matrix $\frac{1}{c}A$ of Example 1 is involutory MDS but not as efficient as the involutory MDS matrix A of Example 2.

Example 3: Here we construct $2^3 \times 2^3$ involutory MDS matrix from Algorithm 2. Let $r = 1$. Using Algorithm 2, we select $a_{0,1} = 02_x$, $a_{0,2} = 06_x$ and $a_{0,4} = 30_x$ of low Hamming weight. This generates $G = \{00_x, 8c_x, 7a_x, f6_x, 2d_x, a1_x, 57_x, db_x\}$. So we generate $r + G$ and finally the involutory MDS matrix A using Algorithm 2, first row of which is as follows: $(01_x \quad 02_x \quad 06_x \quad 8c_x \quad 30_x \quad fb_x \quad 87_x \quad c4_x)$.

Example 4: Here we construct $2^4 \times 2^4$ involutory MDS matrix from Algorithm 2. Let $r = 1$. Using Algorithm 2, we select $a_{0,1} = 03_x$, $a_{0,2} = 08_x$ and $a_{0,4} = 0d_x$ and

$a_{0,8} = 0f_x$ of low Hamming weight. This generates $G = \{00_x, f7_x, e9_x, 1e_x, e0_x, 17_x, 09_x, fe_x, c6_x, 31_x, 2f_x, d8_x, 26_x, d1_x, cf_x, 38_x\}$. So we generate $r + G$ and finally the involutory MDS matrix A using Algorithm 2 which is as follows:

$$A = \begin{pmatrix}
01_x & 03_x & 08_x & b2_x & 0d_x & 60_x & e8_x & 1c_x & 0f_x & 2c_x & a2_x & 8b_x & c9_x & 7a_x & ac_x & 35_x \\
03_x & 01_x & b2_x & 08_x & 60_x & 0d_x & 1c_x & e8_x & 2c_x & 0f_x & 8b_x & a2_x & 7a_x & c9_x & 35_x & ac_x \\
08_x & b2_x & 01_x & 03_x & e8_x & 1c_x & 0d_x & 60_x & a2_x & 8b_x & 0f_x & 2c_x & ac_x & 35_x & c9_x & 7a_x \\
b2_x & 08_x & 03_x & 01_x & 1c_x & e8_x & 60_x & 0d_x & 8b_x & a2_x & 2c_x & 0f_x & 35_x & ac_x & 7a_x & c9_x \\
0d_x & 60_x & e8_x & 1c_x & 01_x & 03_x & 08_x & b2_x & c9_x & 7a_x & ac_x & 35_x & 0f_x & 2c_x & a2_x & 8b_x \\
60_x & 0d_x & 1c_x & e8_x & 03_x & 01_x & b2_x & 08_x & 7a_x & c9_x & 35_x & ac_x & 2c_x & 0f_x & 8b_x & a2_x \\
e8_x & 1c_x & 0d_x & 60_x & 08_x & b2_x & 01_x & 03_x & ac_x & 35_x & c9_x & 7a_x & a2_x & 8b_x & 0f_x & 2c_x \\
1c_x & e8_x & 60_x & 0d_x & b2_x & 08_x & 03_x & 01_x & 35_x & ac_x & 7a_x & c9_x & 8b_x & a2_x & 2c_x & 0f_x \\
0f_x & 2c_x & a2_x & 8b_x & c9_x & 7a_x & ac_x & 35_x & 01_x & 03_x & 08_x & b2_x & 0d_x & 60_x & e8_x & 1c_x \\
2c_x & 0f_x & 8b_x & a2_x & 7a_x & c9_x & 35_x & ac_x & 03_x & 01_x & b2_x & 08_x & 60_x & 0d_x & 1c_x & e8_x \\
a2_x & 8b_x & 0f_x & 2c_x & ac_x & 35_x & c9_x & 7a_x & 08_x & b2_x & 01_x & 03_x & e8_x & 1c_x & 0d_x & 60_x \\
8b_x & a2_x & 2c_x & 0f_x & 35_x & ac_x & 7a_x & c9_x & b2_x & 08_x & 03_x & 01_x & 1c_x & e8_x & 60_x & 0d_x \\
c9_x & 7a_x & ac_x & 35_x & 0f_x & 2c_x & a2_x & 8b_x & 0d_x & 60_x & e8_x & 1c_x & 01_x & 03_x & 08_x & b2_x \\
7a_x & c9_x & 35_x & ac_x & 2c_x & 0f_x & 8b_x & a2_x & 60_x & 0d_x & 1c_x & e8_x & 03_x & 01_x & b2_x & 08_x \\
ac_x & 35_x & c9_x & 7a_x & a2_x & 8b_x & 0f_x & 2c_x & e8_x & 1c_x & 0d_x & 60_x & 08_x & b2_x & 01_x & 03_x \\
35_x & ac_x & 7a_x & c9_x & 8b_x & a2_x & 2c_x & 0f_x & 1c_x & e8_x & 60_x & 0d_x & b2_x & 08_x & 03_x & 01_x
\end{pmatrix}.$$

Example 5: Here we construct $2^5 \times 2^5$ involutory MDS matrix from Algorithm 2. Let $r = 1$. Using Algorithm 2, we select $a_{0,1} = 02_x$, $a_{0,2} = 04_x$ and $a_{0,4} = 07_x$ and $a_{0,8} = 0b_x$ and $a_{0,16} = 0e$ of low Hamming weight. This generates $G = \{00_x, 8c_x, ca_x, 46_x, d0_x, 5c_x, 1a_x, 96_x, c1_x, 4d_x, 0b_x, 87_x, 11_x, 9d_x, db_x, 57_x, e4_x, 68_x, 2e_x, a2_x, 34_x, b8_x, fe_x, 72_x, 25_x, a9_x, ef_x, 63_x, f5_x, 79_x, 3f_x, b3_x\}$. So we generate $r + G$ and finally the involutory MDS matrix A using Algorithm 2, first row of which is as follows: $(01_x\ 02_x\ 04_x\ 69_x\ 07_x\ ec_x\ cc_x\ 72_x\ 0b_x\ 54_x\ 29_x\ be_x\ 74_x\ f9_x\ c4_x\ 87_x\ 0e_x\ 47_x\ c2_x\ c3_x\ 39_x\ 8e_x\ 1c_x\ 85_x\ 55_x\ 26_x\ 1e_x\ af_x\ 68_x\ b6_x\ 59_x\ 1f_x)$. Note that matrices in Example 2 to Example 5 are FFHadamard (see Remark 4). So $h_{i,j} = h_{0,i\oplus j}$ for all $i, j \in \{0, \ldots, 31\}$.

4 FFHadamard MDS Matrices from Cauchy Based Construction and Vandermonde Based Constructions

The authors of [19] constructed FFHadamard involutory MDS matrices starting from two Special Vandermonde matrices. In this section we first show that Cauchy construction of Algorithm 1 gives FFHadamard matrices. We next show (see Theorem 5) the equivalence of Cauchy based construction and Vandermonde based construction of "FFHadamard involutory MDS matrices" of [19]. In doing so, we provide a much simpler proof (see Corollary 8) of generalization of Corollary 2 of [19]. We also prove that Cauchy based construction using Algorithm 1 is faster than the Vandermonde based construction.In the following theorem we show that the MDS matrices constructed by Algorithm 1 are FFHadamard.

Theorem 4. *Algorithm 1 generates FFHadamard Matrices.*

Proof. Let us assume that Algorithm 1 produces $2^m \times 2^m$ matrix $A = ((a_{i,j}))$. So $a_{i,j} = \frac{1}{x_i + y_j} = \frac{1}{r + x_i + x_j} = \frac{1}{r + x_{i\oplus j}}$, where x_i's and y_j's are as defined in the Algorithm 1. From Fact 9, A is FFHadamard matrix. □

4.1 Equivalence of Cauchy Based Construction and Vandermonde Based Construction of Involutory MDS FFHadamard Matrices

Here we fix certain notations that will be used freely in the rest of this Section. Let $\mathbb{G} = \{\gamma_0, \gamma_1, \ldots, \gamma_{d-1}\}$ be an additive subgroup of $\mathbb{F}_{2^n}$ of order d where $\gamma_0 = 0$ and $\gamma_i + \gamma_j = \gamma_{i \oplus j}$ for $i, j \in \{0, \ldots, d-1\}$. For any two arbitrary $r_1, r_2 \in \mathbb{F}_{2^n}$, such that $r_1 + r_2 \notin \mathbb{G}$, let us define three cosets of $\mathbb{G}$ as follows: $r_1 + \mathbb{G} = \{\alpha_i : \alpha_i = r_1 + \gamma_i \text{ for } i = 0, \ldots, d-1\}$, $r_2 + \mathbb{G} = \{\beta_i : \beta_i = r_2 + \gamma_i \text{ for } i = 0, \ldots, d-1\}$ and $r_1 + r_2 + \mathbb{G} = \{\delta_i : \delta_i = r_1 + r_2 + \gamma_i \text{ for } i = 0, \ldots, d-1\}$. Let γ be the product of all nonzero elements of $\mathbb{G}$, β be the product of all elements of $r_2 + \mathbb{G}$ and δ be the product of all elements of $r_1 + r_2 + \mathbb{G}$, i.e. $\gamma = \prod_{k=1}^{d-1} \gamma_k$, $\beta = \prod_{k=0}^{d-1} \beta_k$ and $\delta = \prod_{k=0}^{d-1} \delta_k$. Also let us define two $d \times d$ Special Vandermonde matrices (SV matrices) A and B as follows: $A = van(\alpha_0, \alpha_1, \ldots, \alpha_{d-1})$ and $B = van(\beta_0, \beta_1, \ldots, \beta_{d-1})$ and let

$$B^{-1} = \begin{pmatrix} b_{0,0} & b_{0,1} & \cdots & b_{0,d-1} \\ b_{1,0} & b_{1,1} & \cdots & b_{1,d-1} \\ \vdots & \vdots & \vdots & \vdots \\ b_{d-1,0} & b_{d-1,1} & \cdots & b_{d-1,d-1} \end{pmatrix}, \text{ where } b_{i,j} \in \mathbb{F}_{2^n}.$$

We will prove in Theorem 5 the equivalence of Vandermonde based constructions (see Subsection 3.1 of [19]) and Cauchy based constructions (see Algorithm 1) of FFHadamard involutory MDS matrices. Before going into the proof, we study few properties of B and B^{-1} in Lemma 9 to Lemma 12.

Lemma 9. *$det(B) = \gamma^{d/2}$.*

Proof. From Fact 7, $det(B) = \prod_{k<l}(\beta_k + \beta_l) = (\prod_{k \neq l}(\beta_k + \beta_l))^{1/2} = (\prod_{k \neq l}(\gamma_k + \gamma_l))^{1/2}$. In the product $\prod_{k \neq l}(\gamma_k + \gamma_l)$, each of the terms $\gamma_1, \ldots, \gamma_{d-1}$ occurs d times. So $\prod_{k \neq l}(\gamma_k + \gamma_l) = \prod_{i=1}^{d-1} \gamma_i^d = \gamma^d$. Therefore $det(B) = \gamma^{d/2}$. □

In the next lemma, we show that the elements of last row of B^{-1} i.e. $b_{d-1,j}$'s for $j = 0, \ldots, d-1$ are equal and independent of j.

Lemma 10. *$b_{d-1,j} = \frac{1}{\gamma}$ for $j = 0, \ldots, d-1$.*

Proof. Let $j \in \{0, 1, \ldots, d-1\}$ be arbitrary. So, $b_{d-1,j} = \frac{det(B')}{det(B)}$. Where

$$B' = \begin{pmatrix} 1 & \beta_0 & \beta_0^2 & \beta_0^3 & \cdots & \beta_0^{d-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \beta_{j-1} & \beta_{j-1}^2 & \beta_{j-1}^3 & \cdots & \beta_{j-1}^{d-2} \\ 1 & \beta_{j+1} & \beta_{j+1}^2 & \beta_{j+1}^3 & \cdots & \beta_{j+1}^{d-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \beta_{d-1} & \beta_{d-1}^2 & \beta_{d-1}^3 & \cdots & \beta_{d-1}^{d-2} \end{pmatrix}.$$

Now $(\prod_{k \neq l}(\gamma_k + \gamma_l))^{1/2} = \prod_{k<l}(\gamma_k + \gamma_l)$ and $\prod_{k \neq l, k, l \neq j}(\gamma_k + \gamma_l) = \frac{\prod_{k \neq l,}(\gamma_k + \gamma_l)}{\prod_{k \neq j}(\gamma_k + \gamma_j) \prod_{l \neq j}(\gamma_j + \gamma_l)} = \frac{\prod_{k \neq l,}(\gamma_k + \gamma_l)}{\prod_{k \neq 0} \gamma_k \prod_{l \neq 0} \gamma_l} = \frac{\gamma^d}{\gamma^2} = \gamma^{d-2}$. Therefore $det(B') = \prod_{k<l, k, l \neq j}(\beta_k + \beta_l) = \prod_{k<l, k, l \neq j}(\gamma_k + \gamma_l) = (\prod_{k \neq l, k, l \neq j}(\gamma_k + \gamma_l))^{1/2} = \gamma^{(d-2)/2}$.

From the relation $B^{-1} = \frac{Adj(B)^t}{det(B)}$, we get $b_{n-1,j} = \frac{det(B')}{det(B)}$. Using Lemma 9, we get $b_{n-1,j} = \frac{det(B')}{det(B)} = \frac{\gamma^{(d-2)/2}}{\gamma^{d/2}} = \frac{1}{\gamma}$. □

Let us define $d-1$ degree polynomials $P_j(x) = \sum_{i=0}^{d-1} b_{i,j}x^i$ for $j = 0, \ldots, d-1$. The coefficients of $P_j(x)$ are the elements of j'th column of B^{-1}. In the next lemma we study the roots of $P_j(x)$.

Lemma 11. *The $d-1$ roots of $P_j(x)$ are $\beta_0, \ldots, \beta_{j-1}, \beta_{j+1}, \ldots, \beta_{d-1}$ for $j = 0, \ldots, d-1$.*

Proof. We know $BB^{-1} = I$, where I is the $d \times d$ identity matrix. The d elements in the j'th column of BB^{-1} are $P_j(\beta_0), P_j(\beta_1), \ldots, P_j(\beta_{d-1})$, of which only j'th element i.e. $P_j(\beta_j)$ is one and the rest $d-1$ are zero. Hence the result follows. □

Corollary 6. *$P_j(x) = \frac{1}{\gamma}\prod_{k\neq j}(x+\beta_k)$ for $j = 0, \ldots, d-1$.*

Proof. From Lemma 11, roots of $P_j(x)$ are $\beta_0, \ldots, \beta_{j-1}, \beta_{j+1}, \ldots, \beta_{d-1}$. Therefore, $P_j(x) = b_{d-1,j}\prod_{k\neq j}(x-\beta_k)$. Since elements are from $\mathbb{F}_{2^n}$ which is of characteristic 2, so $P_j(x) = b_{d-1,j}\prod_{k\neq j}(x+\beta_k)$. Also from Lemma 10, $b_{d-1,j} = \frac{1}{\gamma}$ for $j = 0, \ldots, d-1$. Hence $P_j(x) = \frac{1}{\gamma}\prod_{k\neq j}(x+\beta_k)$ for $j = 0, \ldots, d-1$. □

Lemma 12. $\sum_j \frac{1}{\beta_j} = \frac{\gamma}{\beta}$.

Proof. We know, $B^{-1}B = I$. So $(0,0)$'th element of $B^{-1}B$ i.e. $\sum_k b_{0,j} = 1$. Using Corollary 6, we have $P_j(0) = \frac{1}{\gamma}\prod_{k\neq j}\beta_k = \frac{\beta}{\gamma\beta_j}$. But $P_j(0) = b_{0,j}$. So $1 = \sum_j b_{0,j} = \sum_j P_j(0) = \sum_j \frac{\beta}{\gamma\beta_j} = \frac{\beta}{\gamma}\sum_j \frac{1}{\beta_j}$. Thus $\sum_j \frac{1}{\beta_j} = \frac{\gamma}{\beta}$. □

Corollary 7. $\sum_j \frac{1}{\delta_j} = \frac{\gamma}{\delta}$.

Now we propose Theorem 5, which shows the equivalence between Cauchy based Construction of FFHadamard matrices (Algorithm 1) and Vandermonde based Construction of FFHadamard matrices [19]. Let $\frac{1}{c}M$ be the involutory MDS matrix produced by Algorithm 1, where $M = ((m_{i,j}))$, $m_{i,j} = \frac{1}{\gamma_i+\delta_j}$ for $i, j \in \{0, 1, \ldots, d-1\}$, $c = \sum_{k=0}^{d-1}\frac{1}{\delta_k}$. Note that in Algorithm 1, if we take G as $\mathbb{G}$, r as $r_1 + r_2$ and set $b_{Involutory} = true$, then Algorithm 1 constructs $\frac{1}{c}M$.

Theorem 5. $AB^{-1} = \frac{1}{c}M$.

Proof. Let $AB^{-1} = ((h_{i,j}))$. Now, the (i,j)'th element of AB^{-1} is $P_j(\alpha_i)$. Using Corollary 6, we have $h_{i,j} = P_j(\alpha_i) = \frac{1}{\gamma}\prod_{k\neq j}(\alpha_i+\beta_k) = \frac{1}{\gamma}\prod_{k\neq j}(r_1+\gamma_i+r_2+\gamma_k) = \frac{1}{\gamma}\frac{\prod_k(r_1+r_2+\gamma_i+\gamma_k)}{(r_1+r_2+\gamma_i+\gamma_j)} = \frac{1}{\gamma}\frac{\prod_k \delta_k}{(\gamma_i+\delta_j)} = \frac{\delta}{\gamma}\frac{1}{(\gamma_i+\delta_j)} = \frac{\delta}{\gamma}m_{i,j}$. Also from Corollary 7, $c = \sum_{k=0}^{d-1}\frac{1}{\delta_k} = \frac{\gamma}{\delta}$. Thus $h_{i,j} = \frac{1}{c}m_{i,j}$. Hence the proof. □

Note that by Lemma 5 (a generalization of Corollary 2 of [19]), AB^{-1} is an FFHadamard involutory MDS matrix. The following corollary gives an alternative proof of Lemma 5.

Corollary 8. *AB^{-1} is FFHadamard involutory MDS matrix.*

Proof. Since $\frac{1}{c}M$ is FFHadamard involutory MDS (from Theorem 3 and Theorem 4), so is AB^{-1} (from Theorem 5). □

4.2 Comparison of Algorithm 1 Based on Cauchy Based Construction, and Vandermonde Based Construction of [19] to Construct FFHadamard Involutory MDS Matrices

From Theorem 3, the time complexity of constructing $d \times d$ FFHadamard involutory MDS matrix $\frac{1}{c}M$ is $O(d^2)$. In the Vandermonde based Construction [19] to construct FFHadamard involutory MDS matrix AB^{-1}, it requires a multiplication of $d \times d$ matrices A and B^{-1} and the time complexity is $O(d^3)$. So, the Algorithm 1 is faster than the Vandermonde based Construction of FFHadamard involutory MDS matrix in [19].

5 The Matrix $M_{16\times16}$ Used in MDS-AES of [10] Is Not MDS

In [10], authors proposed 16×16 involutory MDS matrix $M_{16\times16}$ by Cauchy based construction with an additional restriction of allowing elements of low Hamming weights. *We checked that their method does not give MDS matrix.* It is easy to verify that the set of inverses of elements of the first row of $M_{16\times16}$ is not a coset of any additive subgroup of $\mathbb{F}_{2^3}$. In fact the authors of [10] did not consider the additive subgroup properly. Some authors [4,15] recommended $M_{16\times16}$ to be used as a diffusion layer, but using this matrix may introduce severe weaknesses. The $M_{16\times16}$ matrix of [10] is given below.

$$M_{16\times16} = \begin{pmatrix}
01_x & 03_x & 04_x & 05_x & 06_x & 07_x & 08_x & 09_x & 0a_x & 0b_x & 0c_x & 0d_x & 0e_x & 10_x & 02_x & 1e_x \\
03_x & 01_x & 05_x & 04_x & 07_x & 06_x & 09_x & 08_x & 0b_x & 0a_x & 0d_x & 0c_x & 10_x & 0e_x & 1e_x & 02_x \\
04_x & 05_x & 01_x & 03_x & 08_x & 09_x & 06_x & 07_x & 0c_x & 0d_x & 0a_x & 0b_x & 02_x & 1e_x & 0e_x & 10_x \\
05_x & 04_x & 03_x & 01_x & 09_x & 08_x & 07_x & 06_x & 0d_x & 0c_x & 0b_x & 0a_x & 1e_x & 02_x & 10_x & 0e_x \\
06_x & 07_x & 08_x & 09_x & 01_x & 03_x & 04_x & 05_x & 0e_x & 10_x & 02_x & 1e_x & 0a_x & 0b_x & 0c_x & 0d_x \\
07_x & 06_x & 09_x & 08_x & 03_x & 01_x & 05_x & 04_x & 10_x & 0e_x & 1e_x & 02_x & 0b_x & 0a_x & 0d_x & 0c_x \\
08_x & 09_x & 06_x & 07_x & 04_x & 05_x & 01_x & 03_x & 02_x & 1e_x & 0e_x & 10_x & 0c_x & 0d_x & 0a_x & 0b_x \\
09_x & 08_x & 07_x & 06_x & 05_x & 04_x & 03_x & 01_x & 1e_x & 02_x & 10_x & 0e_x & 0d_x & 0c_x & 0b_x & 0a_x \\
0a_x & 0b_x & 0c_x & 0d_x & 0e_x & 10_x & 02_x & 1e_x & 01_x & 03_x & 04_x & 05_x & 06_x & 07_x & 08_x & 09_x \\
0b_x & 0a_x & 0d_x & 0c_x & 10_x & 0e_x & 1e_x & 02_x & 03_x & 01_x & 05_x & 04_x & 07_x & 06_x & 09_x & 08_x \\
0c_x & 0d_x & 0a_x & 0b_x & 02_x & 1e_x & 0e_x & 10_x & 04_x & 05_x & 01_x & 03_x & 08_x & 09_x & 06_x & 07_x \\
0d_x & 0c_x & 0b_x & 0a_x & 1e_x & 02_x & 10_x & 0e_x & 05_x & 04_x & 03_x & 01_x & 09_x & 08_x & 07_x & 06_x \\
0e_x & 10_x & 02_x & 1e_x & 0a_x & 0b_x & 0c_x & 0d_x & 06_x & 07_x & 08_x & 09_x & 01_x & 03_x & 04_x & 05_x \\
10_x & 0e_x & 1e_x & 02_x & 0b_x & 0a_x & 0d_x & 0c_x & 07_x & 06_x & 09_x & 08_x & 03_x & 01_x & 05_x & 04_x \\
02_x & 1e_x & 0e_x & 10_x & 0c_x & 0d_x & 0a_x & 0b_x & 08_x & 09_x & 06_x & 07_x & 04_x & 05_x & 01_x & 03_x \\
1e_x & 02_x & 10_x & 0e_x & 0d_x & 0c_x & 0b_x & 0a_x & 09_x & 08_x & 07_x & 06_x & 05_x & 04_x & 03_x & 01_x
\end{pmatrix}.$$

The elements of $M_{16\times16}$ are from $\mathbb{F}_{2^8}$ and the constructing polynomial is $x^8 + x^4 + x^3 + x + 1$. Let us consider the 2×2 submatrix A of $M_{16\times16}$ formed by taking 0th and 2nd row and 1st and 5th column. Let α be the root of $x^8 + x^4 + x^3 + x + 1$. Then in polynomial representation,

$$A = \begin{pmatrix} 03_x & 07_x \\ 05_x & 09_x \end{pmatrix} = \begin{pmatrix} 1+\alpha & 1+\alpha+\alpha^2 \\ 1+\alpha^2 & 1+\alpha^3 \end{pmatrix}.$$

So $det(A) = (1+\alpha)(1+\alpha^3) + (1+\alpha+\alpha^2)(1+\alpha^2) = 1+\alpha^4+\alpha+\alpha^3+1+\alpha^2+\alpha+\alpha^3+\alpha^2+\alpha^4 = 0$. Thus the submatrix A is singular. So clearly from Fact 1, $M_{16\times16}$ is non MDS. Example 4 provides 16×16 involutory MDS matrix which can be used instead of $M_{16\times16}$ of [10]. Note that the matrix in Example 4 does not look as good as $M_{16\times16}$, in terms of Hamming weights of its elements – but $M_{16\times16}$ is non MDS. One can also generate different involutory MDS matrices using Algorithm 2.

6 Conclusion

In this paper, we developed techniques to construct $d \times d$ MDS matrices over $\mathbb{F}_{2^n}$. We proposed a simple algorithm (Algorithm 1) based on Lemma 6. This algorithm is a generalization of the construction proposed in [30]. We propose another algorithm (Algorithm 2) which uses Algorithm 1 iteratively to find software efficient involutory MDS matrices. We find the interesting equivalence of Cauchy based construction (Algorithm 1) and Vandermonde based construction of FFHadamard involutory MDS matrices [19]. We also prove that Cauchy based construction (Algorithm 1) is faster in terms of time complexity compared to Vandermonde based construction of FFHadamard involutory MDS matrices [19]. We have shown that the 16×16 matrix $M_{16\times16}$, used in MDS-AES of [10], is not MDS.

References

1. Barreto, P., Rijmen, V.: The Khazad Legacy-Level Block Cipher, Submission to the NESSIE Project (2000), `http://cryptonessie.org`
2. Barreto, P.S., Rijmen, V.: The Anubis block cipher, NESSIE Algorithm Submission (2000), `http://cryptonessie.org`
3. Bosma, W., Cannon, J., Playoust, C.: The Magma Algebra System I: The User Language. J. Symbolic Comput. 24(3-4), 235–265 (1997); Computational algebra and number theory (London, 1993)
4. Choy, J., Yap, H., Khoo, K., Guo, J., Peyrin, T., Poschmann, A., Tan, C.H.: SPN-Hash: Improving the Provable Resistance against Differential Collision Attacks. In: Mitrokotsa, A., Vaudenay, S. (eds.) AFRICACRYPT 2012. LNCS, vol. 7374, pp. 270–286. Springer, Heidelberg (2012)
5. Daemen, J., Knudsen, L.R., Rijmen, V.: The block cipher SQUARE. In: Biham, E. (ed.) FSE 1997. LNCS, vol. 1267, pp. 149–165. Springer, Heidelberg (1997)
6. Daemen, J., Rijmen, V.: The Design of Rijndael:AES - The Advanced Encryption Standard. Springer (2002)
7. Filho, G.D., Barreto, P., Rijmen, V.: The Maelstrom-0 Hash Function. In: Proceedings of the 6th Brazilian Symposium on Information and Computer Systems Security (2006)
8. Gauravaram, P., Knudsen, L.R., Matusiewicz, K., Mendel, F., Rechberger, C., Schlaffer, M., Thomsen, S.: Grϕstl a SHA-3 Candidate. Submission to NIST (2008), `http://www.groestl.info`
9. Guo, J., Peyrin, T., Poschmann, A.: The `PHOTON` Family of Lightweight Hash Functions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 222–239. Springer, Heidelberg (2011)
10. Nakahara Jr., J., Abrahao, E.: A New Involutory MDS Matrix for the AES. International Journal of Network Security 9(2), 109–116 (2009)
11. Junod, P., Vaudenay, S.: Perfect Diffusion Primitives for Block Ciphers Building Efficient MDS Matrices. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, pp. 84–99. Springer, Heidelberg (2004)
12. Junod, P., Vaudenay, S.: FOX: A new family of block ciphers. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, pp. 114–129. Springer, Heidelberg (2004)

13. Junod, P., Macchetti, M.: Revisiting the IDEA philosophy. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 277–295. Springer, Heidelberg (2009)
14. Lacan, J., Fimes, J.: Systematic MDS erasure codes based on vandermonde matrices. IEEE Trans. Commun. Lett. 8(9), 570–572 (2004)
15. Lo, J.W., Hwang, M.S., Liu, C.H.: An efficient key assignment scheme for access control in a large leaf class hierarchy. Journal of Information Sciences: An International Journal Archive 181(4), 917–925 (2011)
16. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error Correcting Codes. North Holland (1986)
17. Rao, A.R., Bhimasankaram, P.: Linear Algebra, 2nd edn. Hindustan Book Agency
18. Rijmen, V., Daemen, J., Preneel, B., Bosselaers, A., Win, E.D.: The cipher SHARK. In: Gollmann, D. (ed.) FSE 1996. LNCS, vol. 1039, pp. 99–112. Springer, Heidelberg (1996)
19. Sajadieh, M., Dakhilalian, M., Mala, H., Omoomi, B.: On construction of involutory MDS matrices from Vandermonde Matrices in $GF(2^q)$. Design, Codes Cryptography, 1–22 (2012)
20. Sajadieh, M., Dakhilalian, M., Mala, H., Sepehrdad, P.: Recursive Diffusion Layers for Block Ciphers and Hash Functions. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 385–401. Springer, Heidelberg (2012)
21. Schneier, B., Kelsey, J., Whiting, D., Wagner, D., Hall, C., Ferguson, N.: Twofish: A 128-bit block cipher. In: The first AES Candidate Conference. National Institute for Standards and Technology (1998)
22. Schneier, B., Kelsey, J., Whiting, D., Wagner, D., Hall, C., Ferguson, N.: The Twofish encryption algorithm. Wiley (1999)
23. Schnorr, C.-P., Vaudenay, S.: Black Box Cryptanalysis of Hash Networks Based on Multipermutations. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 47–57. Springer, Heidelberg (1995)
24. Shannon, C.E.: Communication Theory of Secrecy Systems. Bell Syst. Technical J. 28, 656–715 (1949)
25. Sony Corporation, The 128-bit Block cipher CLEFIA Algorithm Specification (2007), `http://www.sony.co.jp/Products/cryptography/clefia/download/data/clefia-spec-1.0.pdf`
26. Vaudenay, S.: On the Need for Multipermutations: Cryptanalysis of MD4 and SAFER. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 286–297. Springer, Heidelberg (1995)
27. Watanabe, D., Furuya, S., Yoshida, H., Takaragi, K., Preneel, B.: A new keystream generator MUGI. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 179–194. Springer, Heidelberg (2002)
28. Youssef, A.M., Tavares, S.E., Heys, H.M.: A New Class of Substitution Permutation Networks. In: Workshop on Selected Areas in Cryptography, SAC 1996, Workshop Record, pp. 132–147 (1996)
29. Wu, S., Wang, M., Wu, W.: Recursive Diffusion Layers for (Lightweight) Block Ciphers and Hash Functions. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 355–371. Springer, Heidelberg (2013)
30. Youssef, A.M., Mister, S., Tavares, S.E.: On the Design of Linear Transformations for Substitution Permutation Encryption Networks. In: Workshop on Selected Areas in Cryptography, SAC 1997, pp. 40–48 (1997)